Neural Networks & Deep Learning

ICP-1

Sreeja Reddy Konda

700756597

GitHub Link: https://github.com/SreejaReddyKonda/Neural-Network-Sreeja/tree/main/Neural%20Networks/ICP-1

Video Link:
https://drive.google.com/file/d/1lz_k9IanpGhXDiK5Q51_fiGKmumu73Y1/view?usp=sharing

1.

```
[2]  # method to print full name by combining firstname and lastname
     def full__Name(fName, lName):
         Full_name = first__Name + " " + last__Name
         print("Your full name is ", Full_name)

     # method to print alternative characters of a string
     def string_alternative(str):
         print("Alternative characters are " + str[::2])

     if __name__ == "__main__":
         first__Name = input("Enter your first Name: ")
         last__Name = input("Enter your last Name: ")
         full__Name(first__Name, last__Name)
         str = input("Enter a string to print alternate characters: ")
         string_alternative(str)
```

Output-

```
Enter your first Name: Sreeja Reddy
Enter your last Name: Konda
Your full name is  Sreeja Reddy Konda
Enter a string to print alternate characters: Sreeja Reddy Konda
Alternative characters are Sej ed od
```

Explanation:

The code consists of two functions:

1. **full__Name(fName, lName)**:
    - Combines the given fName (first name) and lName (last name) with a space in between.
    - Prints the full name.
2. **string_alternative(str)**:
    - Takes a string str and extracts every second character using slicing (str[::2]).
    - Prints the alternate characters.

Main Program:

- Prompts the user to input their first and last names, then calls full__Name to print the combined full name.
- Prompts the user for a string and calls string_alternative to print the alternate characters.

2.

```python
f = open("testinput.txt","w")
f.write("Python Course\n")
f.write("Deep learning course\n")
f.close()
f= open("testinput.txt","r")
print(f.read())

from collections import Counter

# Reading input from input.txt
with open('testinput.txt', 'r') as file:
    lines = file.readlines()

# Processing each line and count words
wordcountperline = []

for line in lines:
    words = line.strip().split()
    wordcountperline.append(Counter(words))

# Printing word counts for each word
print("  Word_Count:")
for word, count in Counter(word for wc in wordcountperline for word in wc).items():
    print(f"{word}: {count}")

# Storing the output in output.txt
with open('testoutput.txt', 'w') as output_file:
    for line in lines:
        output_file.write(line)
    output_file.write("Word_Count:\n")
    for word, count in Counter(word for wc in wordcountperline for word in wc).items():
        output_file.write(f"{word}: {count}\n")
```

Output-

```
Python Course
Deep learning course

  Word_Count:
Python: 1
Course: 1
Deep: 1
learning: 1
course: 1
```

Explanation:

File Operations:

- open("testinput.txt", "w"): Opens or creates testinput.txt in write mode, allowing data to be written to the file.
- f.write(): Writes lines of text to the file.
- f.close(): Closes the file, ensuring all data is saved.
- open("testinput.txt", "r"): Reopens the file in read mode to read its contents.
- f.read(): Reads and prints the entire content of the file.

Word Counting:

- from collections import Counter: Imports Counter, a dictionary subclass from the collections module that counts hashable objects like words.
- file.readlines(): Reads all lines from testinput.txt into a list of strings.
- line.strip().split(): Removes leading/trailing whitespace from a line and splits it into individual words.
- Counter(words): Creates a dictionary-like object where each word is a key, and its count is the value.
- wordcountperline.append(): Adds the word count for each line to the wordcountperline list.

Word Count Aggregation:

- Counter(word for wc in wordcountperline for word in wc): Combines the word counts from all lines into a single Counter object, tallying the occurrences of each word across all lines.

Output to a File:

- The word counts are printed and then written to a new file testoutput.txt, appending the word counts below the original text.

3.

```python
import ast

def centimeters_to_inches(centimeters):
    return centimeters / 2.54

# Function to read a list of heights from user input
def get_heights():
    input_string = input("Enter a list of heights in centimeters: ")
    try:
        # Safely evaluate the input string to a list
        heights = ast.literal_eval(input_string)
        if isinstance(heights, list) and all(isinstance(height, int) for height in heights):
            return heights
        else:
            raise ValueError
    except (ValueError, SyntaxError):
        print("Invalid input. Please enter a valid list of integers.")
        return []

# Read heights from user
heights_cm = get_heights()

# Convert to inches using a nested loop
heights_in_inches_loop = []
for height in heights_cm:
    inches = centimeters_to_inches(height)
    heights_in_inches_loop.append(round(inches, 2))

# Convert to inches using list comprehension
heights_in_inches_comprehension = [round(centimeters_to_inches(height), 2) for height in heights_cm]

# Output
print("1. Heights in Inches (Nested Loop):", heights_in_inches_loop)
print("2. Heights in Inches (List Comprehension):", heights_in_inches_comprehension)
```

Output-

```
Enter a list of heights in centimeters: [120, 130, 140]
1. Heights in Inches (Nested Loop): [47.24, 51.18, 55.12]
2. Heights in Inches (List Comprehension): [47.24, 51.18, 55.12]
```

Explanation:

centimeters_to_inches(centimeters) Function: Converts a height from centimeters to inches. It uses the conversion factor where 1-inch equals 2.54 centimeters.

get_heights() Function:

- Prompts the user to enter a list of heights in centimeters as a string.

- Uses ast.literal_eval to safely evaluate the string into a Python list.
- Checks if the result is a list of integers. If not, it prints an error message and returns an empty list.

Conversion Using a Loop:

- Iterates over the list of heights.
- Converts each height to inches using centimeters_to_inches.
- Rounds the result to two decimal places and appends it to a list (heights_in_inches_loop).

Conversion Using List Comprehension:

- Converts the heights to inches using a single line of code that performs the same operations as the loop.
- The result is stored in heights_in_inches_comprehension.

Output:

- Prints the heights in inches obtained from both the nested loop and the list comprehension methods.